REMARKS

Claims 1-15 are pending in the application. Claims 1, 5, 7, 11 and 13-15 are amended herein. Claims 5, 7 and 13 are amended only to correct informalities.

The present invention discloses a novel parallel execution processing system that forms processors into groups according to the requirements of an instruction sequence received by the system for processing. The instruction sequence includes group number information that tells the system how to group the processors so that each group of processors processes only one instruction in the instruction sequence. Each group processes its instruction in parallel with instructions processed by all other groups. This feature allows the present invention to reduce the number of instructions that need to be retrieved for parallel execution. As a result, the time required to retrieve instructions is reduced, and processing speed is advantageously increased. None of the prior art of record has the flexibility to divide a plurality of processors into processing groups for optimizing instruction retrieval time according to the requirements of the instruction received.

To emphasize more clearly patentable features of the present invention, Applicant amends claims 1, 11, 14 and 15 as shown herein in the Listing of Claims. In particular, the present invention as recited in amended Claim 1 provides, *inter alia*, that an instruction sequence obtained by an obtaining unit is decoded into one or more instructions to be assigned to the groups of the processing elements. Thus the processing elements (or processors) are formed into groups and each group is assigned a task of processing a different instruction in the instruction sequence. This permits each group of processors to operate in parallel, and to execute one instruction assigned to the group. In other words, there is one instruction per group, and the instruction is processed in parallel by the processors in the group. Furthermore, the same

obtaining unit also obtains group number information that indicates into how many groups the processors need to be divided.

In the Office Action, Claims 1-7 and 10-15 stand rejected under 35 USC § 103(a) as obvious over USP 4,825,360 ("Knight") in view of USP 5,297,281 ("Emma"). Claims 8-9 stand rejected under 35 USC § 103(a) as obvious over Knight in view of Emma and in further view of J. Rymarczyk, "Coding Guidelines for Pipelined Processors," IBM Corp., 1982 ("Rymarcyzk").

According to MPEP § 706.02(j), for a claim to be obvious, there must be: (a) a suggestion or motivation to combine reference teachings; (b) a reasonable expectation of success; and (c) the references must teach all of the claim limitations. *In re Vaeck*, 947 F.2d 488, 20 U.S.P.Q.2d 1438 (Fed. Cir. 1991). In the present case, the obviousness rejections are unsustainable because no combination of Knight, Emma and/or Rymarczyk teaches all limitations of any independent claim.

The distinguishing feature of the present invention which forms processors into parallel groups wherein each group processes a different instruction is a feature that is absent from the prior art of record. The primary references Knight and Emma both use a technique for dividing an instruction into a plurality of blocks (or groups) and executing the blocks separately:

Knight uses compiler 9 to divide an instruction sequence (or program) into a plurality of blocks (Block 1, Block 2, Block 3). *See* Knight, FIG. 3. Compiler 9 divides these blocks such that each block ends with a side-effecting instruction. Each block is then assigned to a different processor to be executed independently, and at the will of the processor. Knight, 4:47, 60. The result of each execution is stored in a cache memory until the results are validated, or confirmed, in a specific order determined according to the side-effecting instructions. Knight, 4:25-55. Only after confirming each of the blocks in the specific sequence can the entire result be stored in main memory. This is unlike parallel processing according to the present invention, because

(i) Knight's processors are not divided into groups, (ii) each block is assigned to a different processor, and (iii) Knight's blocks are processed at different times.

Turning now to Emma, Emma uses a group dispatcher 120 to divide a single sequence of instructions into groups of sequential instructions, each for execution by a single processor. *See* Emma, 4:35-37; FIG. 2C. That is, one of the instruction groups is assigned to main pipelined processor 124 and the next instruction group is assigned to auxiliary pipelined processor 126. Emma, 5:28-32. The remaining groups are then processed in sequence by the pipelined processors. Thus, each instruction group is assigned to a single processor.

Clearly then, the present invention includes a feature not taught or suggested by Knight or Emma, or any combination of the two. Namely, assigning instructions such that each group of processors receives only one instruction. This feature allows the present invention to reduce the number of instructions that need to be retrieved, resulting in reduced retrieval time and increased processing speed. For example, FIG. 1 of the Application illustrates that four processing elements (130a-130d) form a first group (130a and 130b) and a second group (130c and 130d). Thus, the number of instructions to be retrieved for parallel execution of the four processing elements is two — one to be assigned to the first group and one to be assigned to the second group. Although this particular example illustrates four processors forming two groups, in a more practical case there may be 128 processors forming two groups. When a relatively large number of processors forms a small number of groups, the number of instructions to be retrieved to cause all processors to execute in parallel is significantly reduced, and thus the advantageous effect achieved by the invention is more notable.

On the other hand, in a system using the teachings of Knight and/or Emma, each processor (or pipeline) is assigned one instruction. That means the system must retrieve as many instructions as processors in order for all processors to execute in parallel. In Knight, each

instruction sequence (Block 1, Block 2, Block 3), when retrieved, is assigned to a single processor (*e.g.* Processor 1a, Processor 1b, Processor 1c) for validation of the block. *See* Knight, 8:32-64, FIG. 3. Consequently, this type of processing has higher retrieval times.

Applicant notes that in the Office Action pp. 4-6, Emma is cited primarily for its teachings related to using group number information. But to further distinguish over the combination of Knight and Emma, recall Applicant's use of group number information as recited in amended Claim 1. Applicant recites an obtaining unit that obtains group number information to determine how many processors should form a group. Application, p.3, ln.5-20; p.36, ln 13 to p. 40, ln. 12. On page 4 of the Office Action, Emma is credited as teaching obtaining group number information for this same purpose, and a supporting citation is given as Emma, 7:25-30.

Applicant questions the finding that Emma teaches the use of group number information as recited by Applicant in Claim 1. Emma discloses a group queue 118 that holds program instructions. Emma, 7:7-10. Queue 118 has registers 212 that include a group number bit field used by controller 210 to *access the register* for purposes of grouping the instructions. Emma, 7:26-28; 7:40 to 8:38. Thus, contrary to the Office Action, Emma does not teach an obtaining unit operable to obtain group number information indicating how many groups the processing elements should be formed into. This element is not taught in Emma or anywhere else among the cited references. Therefore, the rejection of Claim 1 under § 103(a) should be withdrawn.

In sum, the present invention is distinguishable from Knight and Emma based on (i) forming processors into groups, (ii) assigning a single instruction to each group, and (iii) obtaining group number information for determining how many processors should form a group. Accordingly, Applicant requests that the Examiner allow Claim 1 as amended, as well as its dependent claims 2-10.

As for the rejections of Claims 8-9 based in part on Rymarcyzk, Applicant contends that these rejections are moot in view of the dependency of claims 8 and 9 on Claim 1.

As for the remaining claims 11-15, Applicant notes that each recites similar distinguishing features as those recited in Claim 1 and discussed above. Applicant therefore requests reconsideration of all claims rejections in view of all of the foregoing remarks.

In view the above, Applicant submits that the present invention is more than adequately distinguished over any combination of the references of record by the presently pending claims, and is worthy of patent protection. Accordingly, Applicant earnestly solicits the Examiner to pass all pending claims to issuance.

If the Examiner believes a telephone interview will assist in the prosecution of this application, the undersigned attorney can be contacted at the listed phone number.

I hereby certify that this correspondence is being transmitted via facsimile to the USPTO at 571-273-8300 on May 8, 2006.

By: Sharon Farnus

Sharon Farnus

Signature

Dated: May 8, 2006

Very truly yours,

SNELL & WILMER L.L.P.

Joseph W. Price
Registration No. 25,124
600 Anton Boulevard, Suite 1400
Costa Mesa, California 92626-7689
Telephone: (714) 427-7420
Facsimile: (714) 427-7799